Faster Algorithms for Mean-Payoff Games *

L. Brim[1], J. Chaloupka[1], L. Doyen[2,4], R. Gentilini[2,3], and J. F. Raskin[2]

[1] Faculty of Informatics, Masaryk University, Brno, Czech Republic
[2] Computer Science Department Université Libre de Bruxelles (U.L.B.), Belgium
[3] Department of Mathematics and Informatics, University of Perugia, Perugia, Italy
[4] LSV, ENS Cachan & CNRS, France
{xchalou1,brim}@fi.muni.cz, raffaella.gentilini@dipmat.unipg.it,{ldoyen,jraskin}@ulb.ac.be

May 10, 2011

Abstract. In this paper, we study algorithmic problems for quantitative models that are motivated by the applications in modeling embedded systems. We consider two-player games played on a weighted graph with mean-payoff objective and with energy constraints. We present a new pseudopolynomial algorithm for solving such games, improving the best known worst-case complexity for pseudopolynomial mean-payoff algorithms. Our algorithm can also be combined with [3] to obtain a randomized procedure with currently the best expected time complexity. The proposed solution relies on a simple fixpoint iteration to solve the log-space equivalent problem of deciding the winner of energy games. Our results imply also that energy games and mean-payoff games can be reduced to safety games in pseudopolynomial time.

* This work is an extended and revised version of [6], appeared in the 5-th Doctoral Workshop on Mathematics and Engineering Methods in Computer Science (MEMICS'09), and [9], appeared in the 3-rd Workshop of the ESF Networking Programme on Games for Design and Verification (GAMES'09).

1 Introduction

Quantitative models Recently, several research efforts have been put in studying quantitative extensions of formalisms like automata and games for modeling quantitative aspects of systems like *embedded systems*. Quantities may represent, for example, the power usage of an embedded component, or the buffer size of a networking element [5].

In this context, Henzinger et al. have studied resource interfaces [5], and more recently, Bouyer et al. have studied weighted (timed) automata and games [4]. In the two papers, the authors consider models where accumulated weight along runs are subject to constraints. For one important variant of those models, the so-called *energy games* (with lower bound constraints), they have proved log-space equivalence to classical mean-payoff games. This log-space equivalence allows to reuse the existing algorithms for solving mean-payoff games.

In this paper, we propose a direct algorithm for solving energy games. Furthermore, using the log-space reduction from mean-payoff games to energy games, we show how our new algorithm for energy games can be used to improve on the existing algorithmic solutions to solve mean-payoff games. In addition to improving the worst-case complexity for solving energy games and mean-payoff games, we believe that our algorithmic solution, which is a fixed point computation, has the potential to be efficiently implemented. We believe that our algorithm is an important step into making the tool support for those quantitative models available and efficient.

Mean-payoff games and energy games Two-player mean-payoff games are played on weighted graphs (in which every edge has an integer weight) with two types of vertices: in player-0 vertices, player 0 chooses the successor vertex from the set of outgoing edges; in player-1 vertices, player 1 chooses the successor vertex from the set of outgoing edges. The game results in an infinite path through the graph. The long-run average of the edge-weights along this path, called the *value* of the play, is won by player 0 and lost by player 1.

The *decision problem* for mean-payoff games asks, given a vertex v and an integer $\nu \in \mathbb{Z}$, if player 0 has a strategy to win a value at least ν when the game starts in v. The associated *strategy synthesis problem* is to construct a strategy for player 0 that ensures a value at least ν, if there exists one. The *three-way partition* problem asks, given a threshold $\nu \in \mathbb{Z}$, to partition the set of vertices of the game into the sets $\langle V_{<\nu}, V_{=\nu}, V_{>\nu}\rangle$, where $V_{<\nu}$ is the subset of vertices from which player 0 can only achieve a value less than ν, $V_{=\nu}$ is the subset of vertices where player 0 can achieve ν but not more, and $V_{>\nu}$ is the subset where player 0 can achieve more than ν. The *value problem* consists in computing the maximal (rational) value that player 0 can achieve from each vertex v of the game. Finally the *(optimal) strategy synthesis problem* is to construct a strategy for player 0 that secures the maximal value.

Mean-payoff games have been first studied by Ehrenfeucht and Mycielski in [1] where it is shown that memoryless (or positional) strategies suffice to achieve the optimal value. This result entails that the decision problem for these games lies in NP $\cap$ coNP [2, 19], and it was later shown to belong to[1] UP $\cap$ coUP [13]. Despite many efforts [20, 19, 16, 7, 8, 22, 12], no polynomial-time algorithm for the mean-payoff game problems is known so far. Beside such a theoretically engaging complexity status, mean-payoff games have plenty of applications, especially in the synthesis, analysis and verification of reactive (non-terminating) systems. Many natural models of such systems include quantitative information, and the corresponding question requires the solution of quantitative games, like mean-payoff games. Concrete examples of applications include various kinds of scheduling, finite-window online string matching, or more generally, analysis of online problems and algorithms, as well as selection with limited storage [19]. Mean-payoff games can even be used for solving the max-plus algebra

[1] The complexity class UP is the class of problems recognizable by unambiguous polynomial time nondeterministic Turing machines [17]. Obviously P$\subseteq$ UP $\cap$ coUP $\subseteq$ NP $\cap$ coNP.

$Ax = Bx$ problem, which in turn has further applications [8]. Beside their applicability to the modeling of quantitative problems, mean-payoff games have tight connections with important problems in game theory and logic. For instance, parity games [11] and the model-checking problem for the modal mu-calculus [15] are poly-time reducible to mean-payoff games [10], and it is a long-standing open question to know whether these problems are in P.

In this paper, we present new algorithmic solutions to the mean-payoff game problems listed above, improving the known upper bounds in terms of worst-case complexity. Our algorithms rely on a reduction to so-called *energy games* [5, 4] that are log-space equivalent to mean-payoff games. In an energy game, given an initial credit c^*, the objective of player 0 is to maintain the sum of the weights (the energy level) positive. The *decision problem* for energy games asks, given a weighted game graph and vertex v, if there exists an initial credit for which player 0 wins from v. It is known that memoryless strategies are sufficient for energy games, and that player 0 essentially needs to ensure that all cycles that can be formed by player 1 have nonnegative weight. We show that energy games can be solved elegantly and efficiently using a notion of *progress measure*. Progress measures for weighted graphs are functions that impose local conditions to ensure global properties of the graph. A notion of *parity* progress measure [21] was exploited in [14] for the algorithmic analysis of parity games. In this paper, we introduce so called *energy* progress measures to witness that all cycles in a graph are nonnegative. We show how to transfer this notion from graphs to games, and we provide an efficient fixpoint algorithm to synthesize a progress measure when it exists. Since energy games are log-space equivalent to mean-payoff games, this also defines a new mean-payoff algorithm which is more elegant and conceptually simpler than the previously known algorithmic solutions.

As we will see below, our procedure to solve the mean-payoff games decision problem achieves a better worst-case complexity than the corresponding best known deterministic pseudopolynomial algorithm due to Zwick and Paterson [19]. Moreover, (optimal) strategies can be synthetized as a (free) byproduct of our algorithm, while [19] requires further computation. Our solution of the mean-payoff value problem is also better than [19] when the maximum weight W in the graph is subexponential, which is the relevant case for comparing pseudopolynomial procedures. Finally, we can combine our deterministic mean-payoff value algorithm with the randomized procedure proposed in [3] to obtain an algorithm with currently the best expected complexity (for all W). We note that in typical applications, where the edge-weights represent, for example, the energy consumption of a physical device, W is usually small in comparison with $|V|$, in which case our deterministic algorithm significatively outperforms—by a linear factor—the previous state-of-the-art solutions, without any use of randomization.

Related works and main results All previous *deterministic* algorithms for mean-payoff games are either pseudopolynomial (i.e., polynomial in the number of vertices $|V|$, the number of edges $|E|$, and the maximal absolute weight W, rather than in the binary representation of W) or exponential [20, 19, 16, 22, 18].

In the late eighties, Gurvich, Karzanov, Khachiyan and Lebedev [20, 2] provided the first (exponential) algorithms for mean-payoff games. Their method, based on the notion of rational price function, was later extended by Pisaruk [16], who considered games with mean-payoff objectives in a slightly more general setting than the one originally proposed by Eherenfeucht and Mycielski [1], and provides a pseudopolynomial upper bound.

The best pseudopolynomial deterministic algorithm for mean-payoff games known so far was designed in 1996 by Zwick and Paterson [19]. They provide a value-iteration algorithm with time complexity $\Theta(|E| \cdot |V|^3 \cdot W)$ for the value problem, and $\Theta(|E| \cdot |V|^2 \cdot W)$ for the decision problem and the three-way partition problem. They also consider the optimal strategy synthesis problem, defining a corresponding $\Theta(|E| \cdot |V|^4 \cdot W \cdot \log \frac{|E|}{|V|})$ pseudopolynomial algorithm.

Algorithms	Problems: Decision Problem 3-Way Partition Problem	Problems: Strategy Synthesis	Note
This paper	$\mathcal{O}(\lvert E\rvert \cdot \lvert V\rvert \cdot W)$	$\mathcal{O}(\lvert E\rvert \cdot \lvert V\rvert \cdot W)$	*Deterministic*
[19]	$\Theta(\lvert E\rvert \cdot \lvert V\rvert^2 \cdot W)$	$\Theta(\lvert E\rvert \cdot \lvert V\rvert^3 \cdot W \cdot log\frac{\lvert E\rvert}{\lvert V\rvert})$	*Deterministic*
[22]	$\mathcal{O}(\lvert E\rvert \cdot \lvert V\rvert \cdot 2^{\lvert V\rvert})$	—	*Deterministic*
[12]	$min(\mathcal{O}(\lvert E\rvert \cdot \lvert V\rvert^2 \cdot W), 2^{\mathcal{O}(\sqrt{\lvert V\rvert \cdot log\lvert V\rvert})})$	$min(\mathcal{O}(\lvert E\rvert \cdot \lvert V\rvert^2 \cdot W)), 2^{\mathcal{O}(\sqrt{\lvert V\rvert \cdot log\lvert V\rvert})})$	*Randomized*

Table 1. Complexity of the main algorithms to solve the mean-payoff game problems 1–3 considered in Section 2.

The best deterministic exponential algorithm for solving mean-payoff games is due to Lifshits and Pavlov [22], who provide a graph decomposition procedure with complexity $\mathcal{O}(|E| \cdot |V| \cdot 2^{|V|})$ for the decision problem, and $\mathcal{O}(|E| \cdot |V| \cdot 2^{|V|} \cdot \log W)$ for the value problem.

In 2007, Björklund and Vorobyov [12] define a randomized algorithm which is both subexponential and pseudopolynomial. Their algorithm solves the decision problem, the three-way partition problem, and the winning strategy synthesis in expected time $\min(\mathcal{O}(|E| \cdot |V|^2 \cdot W), 2^{\mathcal{O}(\sqrt{|V| \cdot \log |V|})})$. For the value problem and the optimal strategy synthesis, the time complexity of their solution is bounded by $\min(\mathcal{O}(|E| \cdot |V|^3 \cdot W \cdot (\log |V| + \log W)), 2^{\mathcal{O}(\sqrt{|V| \cdot \log |V|})} \cdot \log W)$. In particular, the pseudopolynomial terms in the upper bounds given by [12] do not require randomization and improve on [19] for the (optimal) strategy synthesis problem, since winning strategies are obtained as a byproduct of the overall computation. In [3], Andersson and Vorobyov proposed a subexponential randomized solution for discounted payoff games, with application to mean-payoff objectives. In particular, [3] solves the value problem for mean-payoff games in expected time $\mathcal{O}(|V|^2 \cdot |E| \cdot e^{2 \cdot \sqrt{|V| \cdot \ln(|E|/\sqrt{|V|})} + O(\sqrt{|V|} + \ln |E|)})$, which improves [12] for large W.

The (deterministic) algorithms proposed in this work to solve mean-payoff games give new pseudopolynomial upper bounds for all problems considered above. In particular, we provide $\mathcal{O}(|E| \cdot |V| \cdot W)$ algorithms for the decision problem and the three-way partition problem, achieving a linear improvement in $|V|$ of the corresponding previous upper bound. We define an algorithm for the value problem with a complexity $\mathcal{O}(|E| \cdot |V|^2 \cdot W \cdot (\log |V| + \log W))$ while the value-iteration algorithm by Zwick and Paterson has complexity $\mathcal{O}(|E| \cdot |V|^3 \cdot W)$. Thus, our procedure performs better when W is polynomial in $|V|$. When W is exponential in $|V|$, the complexity of both algorithms is outperformed by the $\mathcal{O}(|E| \cdot |V| \cdot 2^{|V|} \cdot \log W)$ algorithm in [22]. Finally, our algorithmic solution for the (optimal) strategy synthesis has complexity $\mathcal{O}(|E| \cdot |V| \cdot W)$ (resp. $\mathcal{O}(|E| \cdot |V|^2 \cdot W(\log |V| + \log W))$), also improving on previous upper bounds in [12].

Tables 1 and 2 summarize the results obtained in this paper and compare them with the main algorithms in the literature.

Structure of the paper The rest of this paper is organized as follows. In Section 2, we provide basic definitions and notations. In Section 3, we develop energy progress measures to be used in Section 4 for solving energy game problems. In Section 5, we build up on the new algorithm to improve the state-of-the-art pseudopolynomial time upper bounds for mean-payoff games.

2 Preliminaries

Weighted graphs Let $\mathbb{Z}$ (resp. $\mathbb{N}$) denote the set of integer (resp. nonnegative integer) numbers. A *weighted graph* $G = (V, E, w)$ consists of a finite set V of vertices, a set $E \subseteq V \times V$ of edges, and a weight function $w : E \to \mathbb{Z}$, assigning integer weights to edges. Given $w : E \to \mathbb{Z}$ and $\nu \in \mathbb{Z}$, we denote by $w - \nu$ the function that assigns to each edge $e \in E$ the weight $w(e) - \nu$. We assume that weighted graphs are *total*, i.e. for all $v \in V$, there exists $v' \in V$ such that $(v, v') \in E$. Given $U \subseteq V$, we denote $E \upharpoonright U$ the restriction of E to U, i.e. $E \upharpoonright U = E \cap U \times U$. Given a function f ranging over V, $f : V \to cod(f)$, and $U \subseteq V$, we denote by $f \upharpoonright U$ the restriction of f to U, i.e. $f \upharpoonright U : U \to cod(f)$ maps each $u \in U$ to $f \upharpoonright U(u) = f(u)$. Given $U \subseteq V$ such that for all $v \in U$, there exists $v' \in U$ with $(v, v') \in E$, we denote by $G \upharpoonright U = (V', E', w')$ the weighted subgraph where $V' = U$, $E' = E \upharpoonright V'$, and $w' = w \upharpoonright V'$. Note that weighted subgraphs are total. A finite path p is a nonempty sequence of vertices $v_0 v_1 \ldots v_n$ such that $(v_i, v_{i+1}) \in E$ for all $0 \leq i < n$. A *cycle* is a finite path $p = v_0 v_1 \ldots v_n$ such that $n \geq 1$ and $v_0 = v_n$. A cycle $v_0 v_1 \ldots v_n$ is *reachable* from v in G if there exists a path $u_0 u_1 \ldots u_m$ in G such that $u_0 = v$ and $u_m = v_0$. The average weight of a cycle $v_0 \ldots v_n$ is equal to $\frac{1}{n} \cdot \sum_{i=0}^{n-1} w(v_i, v_{i+1})$. A path $v_0 v_1 \ldots v_n$ is *acyclic* if $v_i \neq v_j$ for all $0 \leq i < j \leq n$. We say that a cycle in a weighted graph is *negative* (resp. *nonnegative*) if the sum of its edge weights is less than 0 (resp. not less than 0). Given a set of vertices $U \subseteq V$, we denote by $\mathsf{pre}(U)$ the set of vertices having a successor in U, i.e. $\mathsf{pre}(U) = \{v \mid \exists v' \in U : (v, v') \in E\}$, and by $\mathsf{post}(U)$ the set of successors of vertices in U, i.e. $\mathsf{post}(U) = \{v \mid \exists v' \in U : (v', v) \in E\}$.

Game graphs A *game graph* is a tuple $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ where $G^\Gamma = (V, E, w)$ is a weighted graph and $\langle V_0, V_1 \rangle$ is a partition of V into the set V_0 of player-0 vertices and the set V_1 of player-1 vertices. An *infinite game* on Γ is played for infinitely many rounds by two players moving a pebble along the edges of the weighted graph G^Γ. In the first round, the pebble is on some vertex $v \in V$. In each round, if the pebble is on a vertex $v \in V_i$ ($i = 0, 1$), then player i chooses an edge $(v, v') \in E$ and the next round starts with the pebble on v'. A *play* in the game graph Γ is an infinite sequence $p = v_0 v_1 \ldots v_n \ldots$ such that $(v_i, v_{i+1}) \in E$ for all $i \geq 0$. A *strategy* for player i ($i = 0, 1$) is a function $\sigma : V^* \cdot V_i \to V$, such that for all finite paths $v_0 v_1 \ldots v_n$ with $v_n \in V_i$, we have $(v_n, \sigma(v_0 v_1 \ldots v_n)) \in E$. We denote by Σ_i ($i = 0, 1$) the set of strategies for player i. A strategy σ for player i is *memoryless* if $\sigma(p) = \sigma(p')$ for all sequences $p = v_0 v_1 \ldots v_n$ and $p' = v'_0 v'_1 \ldots v'_m$ such that $v_n = v'_m$. We denote by Σ_i^M the set of memoryless strategies of player i. A play $v_0 v_1 \ldots v_n \ldots$ is *consistent* with a strategy σ for player i if $v_{j+1} = \sigma(v_0 v_1 \ldots v_j)$ for all positions $j \geq 0$ such that $v_j \in V_i$. Given an initial vertex $v \in V$, the *outcome* of two strategies $\sigma_1 \in \Sigma_1$ and $\sigma_2 \in \Sigma_2$ in v is the (unique) play $\mathsf{outcome}^\Gamma(v, \sigma_0, \sigma_1)$ that starts in v and is consistent with both σ_0 and σ_1. Given a

	Problems		
Algorithms	***Value Problem***	***Optimal Strategy Synthesis***	**Note**
This paper	$\mathcal{O}(\lvert E\rvert \cdot \lvert V\rvert^2 \cdot W \cdot (log\lvert V\rvert + log W))$	$\mathcal{O}(\lvert E\rvert \cdot \lvert V\rvert^2 \cdot W \cdot (log\lvert V\rvert + log W))$	*Det.*
[19]	$\Theta(\lvert E\rvert \cdot \lvert V\rvert^3 \cdot W)$	$\Theta(\lvert E\rvert \cdot \lvert V\rvert^4 \cdot W \cdot log \frac{\lvert E\rvert}{\lvert V\rvert})$	*Det.*
[22]	$\mathcal{O}(\lvert E\rvert \cdot \lvert V\rvert \cdot 2^{\lvert V\rvert} \cdot log W)$	—	*Det.*
[12]	$min(\mathcal{O}(\lvert E\rvert \cdot \lvert V\rvert^3 \cdot W \cdot (log V + log W)), 2^{\mathcal{O}(\sqrt{\lvert V\rvert \cdot log\lvert V\rvert})} \cdot log W)$	$min(\mathcal{O}(\lvert E\rvert \cdot \lvert V\rvert^3 \cdot W \cdot (log\lvert V\rvert + log W)), 2^{\mathcal{O}(\sqrt{\lvert V\rvert \cdot log\lvert V\rvert})} \cdot log W)$	*Rand.*
[3]	$\mathcal{O}(\lvert V\rvert^2 \cdot \lvert E\rvert \cdot e^{2 \cdot \sqrt{\lvert V\rvert \cdot \ln(\lvert E\rvert / \sqrt{\lvert V\rvert})} + O(\sqrt{\lvert V\rvert} + \ln\lvert E\rvert)})$	—	*Rand.*

Table 2. Complexity of the main algorithms to solve the mean-payoff game problems 4–5 considered in Section 2.

memoryless strategy π_i for player i in the game Γ, we denote by $G^\Gamma(\pi_i) = (V, E_{\pi_i}, w)$ the weighted graph obtained by removing from G^Γ all edges (v, v') such that $v \in V_i$ and $v' \neq \pi_i(v)$.

Mean-Payoff Games [1] A *mean-payoff game* (MPG) is an infinite game played on a game graph Γ where player 0 wins a payoff value defined as the long-run average weights of the play, while player 1 loses that value. Formally, the payoff value of a play $v_0 v_1 \dots v_n \dots$ in Γ is

$$\mathsf{MP}(v_0 v_1 \dots v_n \dots) = \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} w(v_i, v_{i+1}).$$

The value *secured* by a strategy $\sigma_0 \in \Sigma_0$ in a vertex v is

$$\mathsf{val}^{\sigma_0}(v) = \inf_{\sigma_1 \in \Sigma_1} \mathsf{MP}(\mathsf{outcome}^\Gamma(v, \sigma_0, \sigma_1))$$

and the *(optimal) value* of a vertex v in a mean-payoff game Γ is

$$\mathsf{val}^\Gamma(v) = \sup_{\sigma_0 \in \Sigma_0} \inf_{\sigma_1 \in \Sigma_1} \mathsf{MP}(\mathsf{outcome}^\Gamma(v, \sigma_0, \sigma_1)).$$

We say that σ_0 is *optimal* if $\mathsf{val}^{\sigma_0}(v) = \mathsf{val}^\Gamma(v)$ for all $v \in V$. Secured value and optimality are defined analogously for strategies of player 1. Ehrenfeucht and Mycielski [1] show that mean-payoff games are *memoryless determined*, i.e., memoryless strategies are sufficient for optimality and the optimal (maximum) value that player 0 can secure is equal to the optimal (minimum) value that player 1 can achieve.

Theorem 1 ([1]). *For all* MPG $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ *and all vertices* $v \in V$*, we have*

$$\mathsf{val}^\Gamma(v) = \sup_{\sigma_0 \in \Sigma_0} \inf_{\sigma_1 \in \Sigma_1} \mathsf{MP}(\mathsf{outcome}^\Gamma(v, \sigma_0, \sigma_1)) = \inf_{\sigma_1 \in \Sigma_1} \sup_{\sigma_0 \in \Sigma_0} \mathsf{MP}(\mathsf{outcome}^\Gamma(v, \sigma_0, \sigma_1)),$$

and there exist memoryless strategies $\pi_0 \in \Sigma_0^M$ *and* $\pi_1 \in \Sigma_1^M$ *such that*

$$\mathsf{val}^\Gamma(v) = \mathsf{val}^{\pi_0}(v) = \mathsf{val}^{\pi_1}(v).$$

Moreover, *uniform* optimal strategies exist for both players, i.e., a unique memoryless strategy can be used to secure the optimal values, independently of the initial vertex [1].

The next lemmas follow from memoryless determinacy of mean-payoff games.

Lemma 1 ([1, 12]). *Let* $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ *be an* MPG*. For all* $\nu \in \mathbb{R}$*, for all memoryless strategies* $\pi_0 \in \Sigma_0^M$ *for player* 0*, and for all vertices* $v \in V$*, the value* $\mathsf{val}^{\pi_0}(v)$ *secured by* π_0 *in* v *is greater than* ν *if and only if all cycles reachable from* v *in the graph* $G^\Gamma(\pi_0)$ *have average weight greater than* ν*.*

Lemma 2 ([1, 22]). *Let* $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ *be a* MPG *and let* $W = \max_{(v,v')\in E} |w(v, v')|$*. For each vertex* $v \in V$*, the optimal value* $\mathsf{val}^\Gamma(v)$ *is a rational number* $\frac{n}{d}$ *such that* $1 \le d \le |V|$ *and* $|n| \le d \cdot W$*.*

We consider the following five classical problems [19, 12] for a MPG $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$:

1. *Decision Problem*. Given a threshold $\nu \in \mathbb{Z}$ and a vertex $v \in V$, decide if $\mathsf{val}^\Gamma(v) \ge \nu$.
2. *Strategy Synthesis*. Given a vertex $v \in V$ and a threshold $\nu \in \mathbb{Z}$ such that $\nu \le \mathsf{val}^\Gamma(v)$, construct a memoryless strategy $\pi_0 \in \Sigma_0^M$ for player 0 such that $\mathsf{val}^{\pi_0}(v) \ge \nu$.
3. *Three-way partition Problem*. Given a integer threshold $\nu \in \mathbb{Z}$, partition the set V into subsets $V_{>\nu}, V_{<\nu}, V_{=\nu}$ of vertices from which player 0 can secure a payoff greater than ν, less than ν, and equal to ν respectively.
4. *Value Problem*. Compute for each vertex $v \in V$ the value[2] $\mathsf{val}^\Gamma(v)$.
5. *Optimal Strategy Synthesis*. Given a vertex v, construct an optimal strategy from v for player 0.

[2] Note that by lemma 2, this value is a rational number.

Energy Games [5, 4] An *energy game* (EG) is an infinite game on the game graph Γ, where the goal of player 0 is to construct an infinite play $v_0 v_1 \dots v_n \dots$ such that for some *initial credit* $c \in \mathbb{N}$:

$$c + \sum_{i=0}^{j} w(v_i, v_{i+1}) \geq 0 \text{ for all } j \geq 0. \tag{1}$$

The quantity $c + \sum_{i=0}^{j-1} w(v_i, v_{i+1})$ is called the *energy level* of the play prefix $v_0 v_1 \dots v_j$. Given a credit c, a play $p = v_0 v_1 \dots$ is *winning* for player 0 if it satisfies (1), otherwise it is winning for player 1. A vertex $v \in V$ is *winning* for player i if there exists an initial credit c and a winning strategy for player i from v for credit c. In the sequel, we denote by W_i the set of winning states for player i. Energy games are memoryless determined [4], i.e. for all $v \in V$, either v is winning for player 0, or v is winning for player 1, and memoryless strategies are sufficient.

Theorem 2 ([4]). *Let $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ be an EG, for all $v \in V$, the following four statements are equivalent:*

- $\exists \sigma_0 \in \Sigma_0 \cdot \forall \sigma_1 \in \Sigma_1 \cdot \mathsf{outcome}^{\Gamma}(v, \sigma_0, \sigma_1)$ *is winning for player 0;*
- $\forall \sigma_1 \in \Sigma_1 \cdot \exists \sigma_0 \in \Sigma_0 \cdot \mathsf{outcome}^{\Gamma}(v, \sigma_0, \sigma_1)$ *is winning for player 0;*
- $\exists \pi_0 \in \Sigma_0^M \cdot \forall \pi_1 \in \Sigma_1^M \cdot \mathsf{outcome}^{\Gamma}(v, \pi_0, \pi_1)$ *is winning for player 0;*
- $\forall \pi_1 \in \Sigma_1^M \cdot \exists \pi_0 \in \Sigma_0^M \cdot \mathsf{outcome}^{\Gamma}(v, \pi_0, \pi_1)$ *is winning for player 0;*

Using the memoryless determinacy of energy games, we can derive the next lemma.

Lemma 3 ([1, 22]). *Let $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ be an EG, for all vertices $v \in V$, for all memoryless strategies $\pi_0 \in \Sigma_0^M$ for player 0, the strategy π_0 is winning from v if and only if all cycles reachable from v in the weighted graph $G^{\Gamma}(\pi_0)$ are nonnegative.*

We consider the following problems for an energy game $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$:

1. *Decision Problem.* Given $v \in V$, decide if v is winning for player 0.
2. *Strategy Synthesis.* Given $v \in V$, if v is winning for player i ($i = 0, 1$), construct a corresponding winning strategy for player i from v.
3. *Partition Problem.* Construct the sets of vertices W_i ($i = 0, 1$) of winning vertices for player i.
4. *Minimum Credit Problem.* For each vertex $v \in W_0$, compute the minimum initial credit $\mathsf{c}^*(v)$ such that there exists a winning strategy σ_0 for player 0.

Using Lemma 1 and Lemma 3, we can relate the decision problems for MPG and EG as follows.

Theorem 3 ([4]). *Let $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ be a game graph. For all thresholds $\nu \in \mathbb{Z}$, for all vertices $v \in V$, player 0 has a strategy in the MPG $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ that secures value at least ν from v if and only if player 0 has a winning strategy in the EG $\Gamma = (V, E, w - \nu, \langle V_0, V_1 \rangle)$ from v.*

Example 1. Consider the mean-payoff game $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ illustrated on the left of Fig. 1, where player 0 (resp. player 1) controls the square (resp. round) vertices. Assume that player 0 wants to ensure a payoff $\nu \geq 1$ from v. To solve such a mean-payoff decision problem we can consider the energy game $\Gamma' = (V, E, w - 1, \langle V_0, V_1 \rangle)$, on the right of Fig. 1, where the weights of all edges are decreased by 1. By construction, each cycle c in the EG Γ' is nonnegative if and only if c has mean-payoff $\nu \geq 1$ in Γ. In particular, player 0 has a strategy to confine the play into the nonnegative cycle (zwz) and win the EG Γ' from v (with initial credit 6). Therefore, player 0 has a strategy to confine the play into the cycle (zwz) having mean-payoff $\nu \geq 1$ in the MPG Γ.

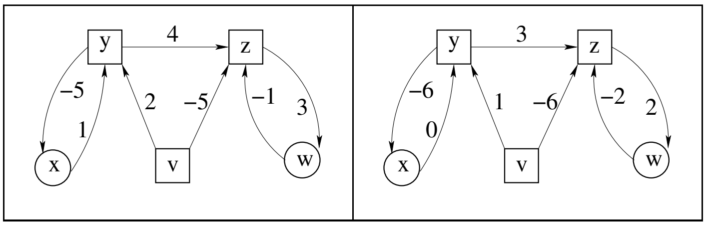

Fig. 1. Solving MPG via EG.

3 A Small Energy Progress Measure

Progress measures are functions $f : V \to \mathbb{N}$, defined *locally* on the set of vertices of a weighted graph, that allow to infer *global* properties of the graph. In this section, we introduce a notion of progress measure called *energy progress measure*, which is tailored to witness the absence of negative cycles in a weighted graph G. Intuitively, the value $f(v_0)$ of a vertex v_0 is a sufficient credit to ensure that all paths $v_0 \dots v_n$ can be traversed while maintaining a nonnegative level.

Definition 1 (Energy Progress Measure). *Let $G = \langle V, E, w\rangle$ be a weighted graph. An* energy progress measure *for G is a function $f : V \to \mathbb{N}$ such that for all $(v, v') \in E$:*

$$f(v) \geq f(v') - w(v, v').$$

Lemma 4. *Let $G = (V, E, w)$ be a weighted graph. If G admits an energy progress measure, then:*

1. *all cycles of G are nonnegative, and*
2. *for all paths $v_0 v_1 \dots v_n$ in G it holds:*

$$f(v_0) + \sum_{i=0}^{n-1} w(v_i, v_{i+1}) \geq 0$$

Proof. Let $G = (V, E, w)$ be a weighted graph and f be an energy progress measure for G. Consider an arbitrary path $p = v_0 v_1 \dots v_n$ in G. By definition of energy progress measure, we have:

$$f(v_0) \geq f(v_1) - w(v_0, v_1) \geq \dots \geq f(v_n) - \sum_{i=0}^{n-1} w(v_i, v_{i+1}). \tag{2}$$

This leads to $f(v_0) + \sum_{i=0}^{n-1} w(v_i, v_{i+1}) \geq f(v_n) \geq 0$, which proves item 2. In the particular case where p is a cycle (i.e., $v_0 = v_n$) Inequality (2) can also be developed into $\sum_{i=0}^{n-1} w(v_i, v_{i+1}) \geq 0$ which proves item 1. □

The next lemma shows that if all cycles of G are nonnegative, then G admits an energy progress measure whose codomain has a pseudopolynomial upper bound (in the size of G). Hence, we refer to our progress measure as a *small* energy progress measure. Given a weighted graph $G = (V, E, w)$, define:

$$\mathcal{M}_G = \sum_{v \in V} \max(\{0\} \cup \{-w(v, v') \mid (v, v') \in E\})$$

Note that, if $W = \max_{e \in E} |w(e)|$ is the maximal absolute value of the edge-weights in G, then $\mathcal{M}_G \leq |V| \cdot W$.

Lemma 5. *For all weighted graphs $G = (V, E, w)$, if all cycles of G are nonnegative, then there exists an energy progress measure $f : V \to \{0, \ldots, \mathcal{M}_G\}$ for G.*

Proof. Given $v \in V$, let $\mathsf{AcyclicPath}(v)$ be the set of (possibly trivial[3]) acyclic paths in $G = (V, E, w)$ starting in v:

$$\begin{aligned}\mathsf{AcyclicPath}(v) = \{v_0 v_1 \ldots v_n \mid v_0 = v \wedge \forall 0 \le i < n : (v_i, v_{i+1}) \in E \\ \wedge \forall 0 \le i, j \le n : i \ne j \to v_i \ne v_j\}\end{aligned}$$

Given $p = v_0 v_1 \ldots v_n \in \mathsf{AcyclicPath}(v)$, we denote by $w(p)$, the sum of the weights in p:

$$w(p) = \begin{cases} 0 & \text{if } n = 0 \\ \sum_{i=0}^{n-1} w(v_i, v_{i+1}) & \text{otherwise} \end{cases}$$

Consider the function $f : V \to \{0, \ldots, \mathcal{M}_G\}$ defined by:

$$f(v) = \max\{-w(p) \mid p \in \mathsf{AcyclicPath}(v)\}$$

for all $v \in V$. Note that by definition of $w(p)$, we have $f(v) \ge 0$. We claim that f is an energy progress measure for G. Towards contradiction, assume that there exists an edge (v, v') for which:

$$f(v) < f(v') - w(v, v'). \tag{3}$$

There are two cases to consider, depending on whether v is equal to v' or not. In the first case ($v = v'$), Inequality 3 immediately yields the contradiction that (v, v) is a negative cycle in G. In the second case ($v \ne v'$), let $p_{v'} = v_0 v_1 \ldots v_n$ be an acyclic path in G from v' (i.e. $v_0 = v'$) such that $f(v') = -w(p_{v'})$. If $p_{v'}$ does not contain v, then by definition of f we get $f(v) \ge -w(p_{v'}) - w(v, v')$ which contradicts Inequality 3. Otherwise, let $0 < i \le n$ such that $v_i = v$, and let $w^1 = w(v_0 v_1 \ldots v_i)$ and $w^2 = w(p_{v'}) - w^1$. By Inequality 3, we have $w(v, v') + w^1 + w^2 < -f(v)$. Since all cycles of G are nonnegative, we have $w(v, v') + w^1 \ge 0$, and thus $w^2 < -f(v)$, i.e. $f(v) < -w^2$. This is again in contradiction with the definition of f since w^2 is the weight of a (possibly trivial) acyclic path from the vertex v. □

4 Solving the Energy Game Problems

In this section, we devise efficient algorithms for the EG problems stated in Section 2. To this purpose, we extend the notion of small progress measure from graphs to games, taking into account the partition of vertices between the two players. Let $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ be a game graph and consider the set:

$$\mathcal{C}_\Gamma = \{n \in \mathbb{N} \mid n \le \mathcal{M}_{G^\Gamma}\} \cup \{\top\}.$$

We denote by $\preceq$ the total order on $\mathcal{C}_\Gamma$ defined by $x \preceq y$ if and only if either $y = \top$ or $x \le y \le \mathcal{M}_{G^\Gamma}$. Moreover, we define the operator $\ominus : \mathcal{C}_\Gamma \times \mathbb{Z} \to \mathcal{C}_\Gamma$ such that, for all $a \in \mathcal{C}_\Gamma$ and $b \in \mathbb{Z}$:

$$a \ominus b = \begin{cases} \max(0, a - b) & \text{if } a \ne \top \text{ and } a - b \le \mathcal{M}_{G^\Gamma} \\ \top & \text{otherwise} \end{cases}$$

Intuitively, a small energy progress measure for the game $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ is a mapping from V to $\mathcal{C}_\Gamma$ tailored to witness wether a vertex v is winning for player 0. In particular, if the small energy progress measure function f assumes a value $f(v) \ne \top$ on the vertex v, then player 0 has a winning strategy from v, provided an initial credit $f(v)$.

[3] For each $v \in V$, there is a trivial acyclic path v in G.

Definition 2. *Let $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$ be an* EG*. A function $f : V \to \mathcal{C}_\Gamma$ is a* small energy progress measure *for Γ if and only if the following conditions hold:*

- *if $v \in V_0$, then $f(v) \succeq f(v') \ominus w(v, v')$ for some $(v, v') \in E$;*
- *if $v \in V_1$, then $f(v) \succeq f(v') \ominus w(v, v')$ for all $(v, v') \in E$.*

Note that Definition 2 can be derived by the corresponding Definition 1 (on graphs) by means of the following two extensions. First, specialize the local conditions constraining the (small) energy progress measure on each node $v \in V$ by taking into account wether $v \in V_0$ or $v \in V_1$. Second, introduce the special value $\top$ in the codomain of the small energy progress measure[4] f, $f : V \mapsto \mathcal{C}_\Gamma = \{0, \ldots, \mathcal{M}_{G_\Gamma}\} \cup \{\top\}$, ensuring that all games admit a small energy progress measure.

Given a small energy progress measure f for the game graph $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$, we denote by V_f the set of states $V_f = \{v \mid f(v) \neq \top\}$. A (memoryless) strategy $\pi_0^f : V_0 \to V$ for player 0 is called *compatible with* f whenever for all $v \in V_0$, if $\pi_0^f(v) = v'$ then $f(v) \succeq f(v') \ominus w(v, v')$. Note that compatible strategies always exist by definition of progress measure. The next lemma establishes that if π_0^f is a strategy for player 0 compatible with the energy progress measure f, then π_0^f is a winning strategy for player 0 from all vertices in V_f.

Lemma 6. *Let $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$ be an* EG*. For all small energy progress measures f for Γ, if π_0^f is a strategy for player 0 compatible with f, then π_0^f is a winning strategy for player 0 from all vertices $v \in V_f$, i.e. $V_f \subseteq W_0$. Moreover, Γ admits a small energy progress measure f such that $V_f = W_0$.*

Proof. Let f be a small energy progress measure for Γ and consider a memoryless strategy π_0^f for player 0 which is compatible with f. For the sake of contradiction, suppose that π_0^f is not winning for player 0 from the vertex $v \in V_f$. Then, by Lemma 3, $G^\Gamma(\pi_0^f)$ admits a negative cycle from v. Let $v_0 v_1 \ldots v_i \ldots v_n$ with $v_0 = v$ be the path in $G^\Gamma(\pi_0^f)$ from v with a negative cycle $v_i \ldots v_n$ (i.e., $v_i = v_n$). We show that $v_j \in V_f$ for all $0 \leq j \leq n$, using an inductive argument on j. The base case is obvious since $v_0 = v \in V_f$ by hypothesis. Let $j > 0$. By inductive hypothesis we have that $v_{j-1} \in V_f$. By definition of small energy progress measure on Γ, if $v_{j-1} \in V_0$ (resp. $v_{j-1} \in V_1$), then there exists a successor (resp. for all successors) v' of v_{j-1}:

$$f(v_{j-1}) \succeq f(v') \ominus w(v_{j-1}, v'). \tag{4}$$

By Inequality 4 and by definition of π_0^f we obtain $f(v_j) \neq \top$, i.e. $v_j \in V_f$.

Hence, for each vertex v_j, $i \leq j \leq n$, on the negative cycle $v_i \ldots v_n = v_i$ reachable from v, $f(v_j) \neq \top$. Thus, by definition of f and π_0^f we obtain:

$$f(v_i) \geq f(v_{i+1}) - w(v_i, v_{i+1}) \geq \cdots \geq f(v_n) - \sum_{j=i}^{n-1} w(v_j, v_{j+1})$$

which is a contradiction with our hypothesis that $v_i \ldots v_n$ is a negative cycle.

We conclude by showing that there exists a small energy progress measure f on Γ such that $V_f = W_0$. Let π_0 be a memoryless strategy winning for player 0 from any vertex $v \in W_0$. By Lemma 3, $G^\Gamma(\pi_0) \upharpoonright W_0$ does not contain any negative cycle. Hence, $G^\Gamma(\pi_0) \upharpoonright W_0$ admits an energy progress measure f by Lemma 5. The function f can immediately be extended to an energy progress measure on the game Γ by setting $f(v) = \top$ for each $v \notin W_0$. □

[4] and appropriately define the operator $\ominus : \mathcal{C}_\Gamma \times \mathbb{Z} \mapsto \mathcal{C}_\Gamma$ in order to cast the minus operator to range over $\mathcal{C}_\Gamma$

For a game graph $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$, let $\mathcal{F}$ be the set of functions $f : V \to \mathcal{C}_\Gamma$. The partial order $\sqsubseteq \subseteq \mathcal{F} \times \mathcal{F}$ is defined as $f \sqsubseteq g$ iff for all $v \in V$, $f(v) \preceq g(v)$. Note that for all functions f and g, if $f(v) \succeq f(v') \ominus w(v, v')$ and $g(v) \succeq g(v') \ominus w(v, v')$, then $\min\{f(v), g(v)\} \succeq \min\{f(v'), g(v')\} \ominus w(v, v')$. Therefore, if f and g are small energy progress measures, then so is the function $h = \min\{f, g\}$ (where $\min$ is taken pointwise). We use $(\mathcal{F}, \sqsubseteq)$ to refer to $\mathcal{F}$ as a complete partial order. Given any set $F \subseteq \mathcal{F}$, we denote by $\sqcap F$ the greatest lower bound of F. As $\mathcal{F}$ is a complete partial order, we know that $\sqcap F \in \mathcal{F}$. We can now state the following two important properties.

Proposition 1. *Let $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ be an* EG*, (i) if f and g are small energy progress measures for Γ such that $f \sqsubseteq g$, then $V_g \subseteq V_f$, and (ii) if $f = \sqcap\{g \in \mathcal{F} \mid g$ is a small energy progress measure for $\Gamma\}$, then f is a small energy progress measure and $V_f = W_0$.*

Proof. The first item is immediate by definition of V_f and $\sqsubseteq$. The second item follows from Lemma 6, item 1 and from the fact that if f and g are small energy progress measures, then so is the function $h = \min\{f, g\}$ (where $\min$ is taken pointwise). □

By Lemma 6 and Proposition 1, the problem of determining the least energy progress measure for the energy game $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ subsumes the decision problem for Γ. Hence, we present here an efficient algorithm (Algorithm 1) to compute the least energy progress measure $f : V \to \mathcal{C}_\Gamma$. Our algorithm initializes f to the constant function 0 and relies on the following operator.

Definition 3. *Given $v \in V$, the lifting operator $\delta(\cdot, v) : [V \to \mathcal{C}_\Gamma] \to [V \to \mathcal{C}_\Gamma]$ is defined by $\delta(f, v) = g$ where:*

$$g(u) = \begin{cases} f(u) & \text{if } u \neq v \\ \min\{f(v') \ominus w(v, v') \mid (v, v') \in E\} & \text{if } u = v \in V_0 \\ \max\{f(v') \ominus w(v, v') \mid (v, v') \in E\} & \text{if } u = v \in V_1 \end{cases}$$

The operator $\delta(\cdot, v)$ can be computed in time $\mathcal{O}(|\mathsf{post}(v)|)$, and is $\sqsubseteq$-monotone.

Lemma 7. *For each $v \in V$, the operator $\delta(\cdot, v)$ is monotone, i.e. $\delta(f, v) \sqsubseteq \delta(g, v)$ for all $f \sqsubseteq g$.*

Proof. Immediate from Definition 3. □

Given a function $f : V \to \mathcal{C}_\Gamma$, we say that f is *inconsistent* in v if:

- $v \in V_0$ and for all v' such that $(v, v') \in E$ it holds $f(v) \prec f(v') \ominus w(v, v')$;
- $v \in V_1$ and there exists v' such that $(v, v') \in E$ and $f(v) \prec f(v') \ominus w(v, v')$.

Algorithm 1 maintains a list L of vertices that witness an inconsistency of f. Initially, $v \in V_0 \cap L$ if and only if all outgoing edges from v are negative, while $v \in V_1 \cap L$ if and only if v is the source of a negative edge. As long as the list L is nonempty, the algorithm picks a vertex u from L and performs the following operations:

1. apply to f the lifting operator $\delta(f, v)$ in order to solve the inconsistency of f in v;
2. insert into the list L the set of vertices witnessing a new inconsistency, due to the increase of $f(v)$.

The update of L following a lifting operation $\delta(f, v)$ requires $\mathcal{O}(|\mathsf{pre}(v)|)$ time. In fact, a vertex v' can witness a new inconsistency because of the incrementing of $f(v)$ only if $v' \in \mathsf{pre}(v)$. In particular, checking if $v' \in \mathsf{pre}(v) \cap V_1$ witnesses a new inconsistency simply amounts at checking whether $f(v') \prec f(v) \ominus w(v', v)$. Some more attention needs to be paid for vertices in $\mathsf{pre}(v) \cap V_0$. Indeed, for such vertices the condition $f(v') \prec f(v) \ominus w(v', v)$ may not be sufficient to witness a new inconsistency in v', due to the existence of another successor v'' of v' such that $f(v') \succeq f(v'') \ominus w(v', v'')$. In order to efficiently determine if v' needs to be inserted in L, we maintain a counter function $\mathtt{count} : V_0 \to \mathbb{N}$ such that $count(v) = 0$ for all $v \in V_0 \cap L$, and $\mathtt{count}(v)$ is the number

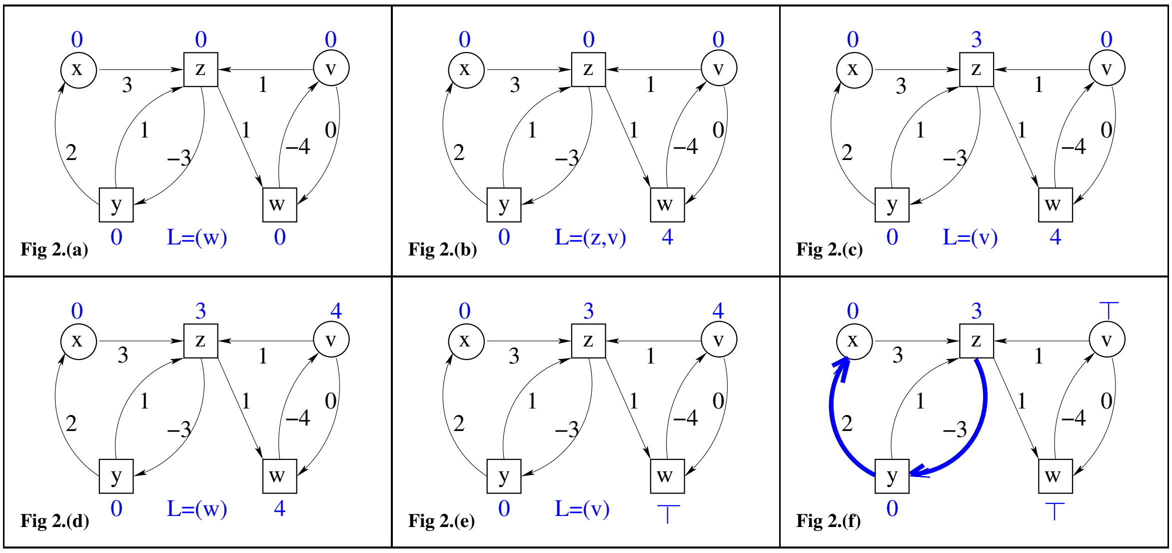

Fig. 2. EG algorithm applied on a concrete game graph, illustrated in Example 2.

of successors v' of v such that $f(v) \succeq f(v') \ominus w(v,v')$ for $v \in V_0 \setminus L$. Initially, $\mathsf{count}(v) \succeq 1$ for all $v \in V_0 \setminus L$. When the value $f(v)$ is updated, we compute the new value of $\mathsf{count}(v)$ (with cost $\mathcal{O}(|\mathsf{post}(v)|)$), and we decrement the value $\mathsf{count}(v')$ of all predecessors v' of v such that $f(v') \prec f(v) \ominus w(v', v)$. Those predecessors v' for which $\mathsf{count}(v')$ is now 0 are inserted in L. The algorithm terminates when the list L is empty. Example 2 illustrates our EG algorithm on a concrete game graph.

Example 2. Consider the game graph $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$ illustrated in Fig. 2.(a), where player 0 (resp. player 1) controls the square (resp. round) vertices. Algorithm 1 initializes the energy progress measure f to the constant function 0, and the list L with the only node w. Fig. 2.(b) shows the result of the execution of the main while-loop at line 7, upon the extraction of the vertex w. In particular, $f(w)$ is updated to 4 leading to the insertion of the nodes z, v into L, within the innermost for-loop. Fig. 2.(c) illustrates the energy progress measure computed by the second iteration of the while-loop at line 7, when z is taken from L. In this case, the new value 3 of $f(z)$ does not lead to any new insertion into L. In fact, at this point of the computation, the values $f(x), f(y), f(z)$ are fixed, and only $f(w)$ and $f(v)$ continue to increase until reaching the maximal value encoded as $\top$. Fig. 2.(e) and 2.(f) show the last steps of the algorithm, and the corresponding winning strategy for player 0.

Note that, once a small energy progress measure f has been computed and $W_0 = V_f$ has been determined, a (memoryless) winning strategy σ_0 for player 0 on W_0 can be immediately derived in time $\mathcal{O}(|E|)$, as follows: For each vertex $v \in V_0$, set $\sigma_0(v) = v'$, where $(v, v') \in E$ and $f(v') = \min\{f(v') \ominus w(v, v') \mid (v, v') \in E\}$. Such a strategy could also be computed online throughout the execution of Algorithm 1, rather than as a post-processing operation.

The correctness of the algorithm is established by Theorem 4 on the ground of Lemma 7 and Lemma 8, applying the Knaster-Tarski fixpoint theorem to our lifting operator in $(\mathcal{F}, \sqsubseteq)$. In particular, the function f computed by Algorithm 1 is a simultaneous least fixpoint of the operators $\delta(\cdot, v)$ for all $v \in V$. Thus, the function f is the least energy progress measure for Γ (since f is the least fixpoint of $\delta(\cdot, v)$ for all $v \in V$) such that $V_f = W_0$ (since f is a *least* fixpoint of $\delta(\cdot, v)$ for all $v \in V$).

Lemma 8. *The following is an invariant of the while-loop of Algorithm 1 (line 7): for all vertices $v \in V \setminus L$, (i) $\delta(f, v) = f$ and (ii) if $v \in V_0$, then* $\mathsf{count}(v) = |\{v' \in V \mid f(v) \succeq f(v') \ominus w(v, v')\}|$.

Algorithm 1: Value-iteration algorithm for energy games.

Input : A game graph $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$.

Output : A small energy progress measure $f : V \to \mathcal{C}_\Gamma$ for Γ.

begin

1 $L \leftarrow \{v \in V_0 \mid \forall (v, v') \in E : w(v, v') < 0\}$

2 $L \leftarrow L \cup \{v \in V_1 \mid \exists (v, v') \in E : w(v, v') < 0\}$

3 **foreach** $v \in V$ **do**

4 $\quad f(v) \leftarrow 0$

5 $\quad$ **if** $v \in V_0 \cap L$ **then** $\mathsf{count}(v) \leftarrow 0$

6 $\quad$ **if** $v \in V_0 \setminus L$ **then** $\mathsf{count}(v) \leftarrow |\{v' \in \mathsf{post}(v) \mid f(v) \succeq f(v') \ominus w(v, v')\}|$

7 **while** $L \neq \emptyset$ **do**

8 $\quad$ Pick $v \in L$

9 $\quad L \leftarrow L \setminus \{v\}$; $old \leftarrow f(v)$

10 $\quad f \leftarrow \delta(f, v)$

11 $\quad$ **if** $v \in V_0$ **then** $\mathsf{count}(v) \leftarrow |\{v' \in \mathsf{post}(v) \mid f(v) \succeq f(v') \ominus w(v, v')\}|$

12 $\quad$ **foreach** $v' \in \mathsf{pre}(v)$ **such that** $f(v') \prec f(v) \ominus w(v', v)$ **do**

13 $\quad\quad$ **if** $v' \in V_0$ **then**

14 $\quad\quad\quad$ **if** $f(v') \succeq old \ominus w(v', v)$ **then** $\mathsf{count}(v') \leftarrow \mathsf{count}(v') - 1$

15 $\quad\quad\quad$ **if** $\mathsf{count}(v') \leq 0$ **then** $L \leftarrow L \cup \{v'\}$

16 $\quad\quad$ **if** $v' \in V_1$ **then** $L \leftarrow L \cup \{v'\}$

17 **return** f

end

Proof. First, we show that the invariant holds after line 6. Consider an arbitrary vertex $v \in V \setminus L$. If $v \in V_0$, then there exists $(v, v') \in E$ such that $w(v, v') \geq 0$ (line 1 of Alg. 1). Since $f(v') = 0$, we get $f(v') \ominus w(v, v') = 0 = f(v)$, showing that $\delta(f, v)(v) = 0 = f(v)$ by Definition 3. It is obvious that $\delta(f, v)(v') = f(v')$ for all $v' \neq v$, and thus $\delta(f, v) = f$. The proof of part (ii) of the invariant is straightforward. The case $v \in V_1$ is proven analogously.

Second, assume that the invariant holds before executing the loop, and let v be the vertex selected at line 8. Consider the case where $(v, v) \notin E$. Let $f' = \delta(f, v)$ (see also line 10). Note that f' differs from f only in the value assigned to vertex v, i.e. $f'(v') = f(v')$ for all $v' \neq v$. Therefore, the value $\mathsf{count}(v')$ needs to be updated only for the predecessors $v' \in \mathsf{pre}(v)$ of v, and this can be done as in line 14. Now, since we assumed that $v \notin \mathsf{pre}(v)$, the vertex v is not inserted back in the list L in the loop of line 12, and thus we need to show that $\delta(f', v) = f'$. It is easy to see that $\delta(f', v)(v') = f'(v')$ for all $v' \neq v$, while for $v' = v$, this follows from the fact that f' and f agree on the value of all successors of v.

Finally, it is easy to see that the list L is correctly updated in lines 12-16: for $v' \in \mathsf{pre}(v) \cap V_1$, if $f(v') \succeq f(v) \ominus w(v', v)$ (i.e. v' is not inserted in the list), then $\delta(f', v') = f'$; for $v' \in \mathsf{pre}(v) \cap V_0$, if the value $\mathsf{count}(v')$ is positive (i.e. v' is not inserted in the list), then $\delta(f', v') = f'$.

The case where $(v, v) \in E$ is proven analogously. □

Theorem 4 (Correctness). *Let $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ be an* EG*. Algorithm 1 computes a small energy progress measure f on Γ such that $V_f = W_0$ is the set of winning vertices for player* 0.

Proof. By Lemma 7 , Lemma 8, and the Knaster-Tarski theorem, the function computed f returned by Algorithm 1 is the unique least fixpoint of simultaneously all operators $\delta(\cdot, v)$ for all $v \in V$. Therefore, the set V_f is the set of winning vertices for player 0 according to Lemma 6.

Termination of Algorithm 1 is enforced by the fact that every update of line 10 strictly increases the value of f in one vertex v, and the fact that the codomain of energy progress measures is finite. □

Theorem 5 characterizes the small energy progress measure computed by Algorithm 1, putting it into relation with the minimum credit problem, and Theorem 6 establishes the complexity of Algorithm 1.

Theorem 5 (Minimal credit). *Let $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$ be an* EG. *The small energy progress measure f computed by Algorithm 1 is such that: (i) if $v \in W_0$ (v is winning for player 0), then $\mathsf{c}^*(v) = f(v)$, where $\mathsf{c}^*(v)$ is the minimum initial credit to build a winning play for player 0 from v. (ii) if $v \in W_1$ (v is winning for player 1), then $f(v) = \top$.*

Proof. By Theorem 4, the function f is a small energy progress measure on Γ such that $V_f = W_0$. Hence, for each $v \notin W_0$, $f(v) = \top$, which establishes the second item of our claim.

In order to show that for each $v \in W_0$, $f(v) = \mathsf{c}^*(v)$, we start proving that f is an energy progress measure on the graph $G^\Gamma(\pi_0^f) \restriction V_f$. This will immediately imply that $f(v) \geq \mathsf{c}^*(v)$ for all $v \in W_0$ by Lemma 4. Let (v, v') be an edge in the graph $G^\Gamma(\pi_0^f) \restriction V_f$. Then $f(v) \neq \top$, $f(v') \neq \top$, and by definition of f we immediately obtain that $f(v) \geq f(v') - w(v, v')$, which yields our claim. We finally get to the result by showing that for each $v \in W_0$, the relation

$$f(v) \leq \mathsf{c}^*(v) \tag{5}$$

is an invariant of Algorithm 1. By contradiction, let $v \in W_0$ be the first node for which Equation 5 is falsified within the execution of Algorithm 1. Since $f(v)$ is initialized to the constant 0, such a violation needs to occur immediately after $f(v)$ gets updated, at line (10), to the value $f(v') - w(v, v')$, for some successor v' of v. Then:

$$f(v') - w(v, v') = f(v) > \mathsf{c}^*(v) \geq \mathsf{c}^*(v') - w(v, v'). \tag{6}$$

Equation 6 implies $f(v') > \mathsf{c}^*(v')$, which contradicts the fact that v was the first node witnessing a violation of equation 5. □

Theorem 6 (Complexity). *The worst-case complexity of Algorithm 1 is $\mathcal{O}(|E| \cdot \mathcal{M}_{G^\Gamma})$.*

Proof. The initialization phase (lines (1)–(6)) costs $\mathcal{O}(\sum_{v \in V}(|\mathsf{post}(v)|)) = \mathcal{O}(|E|)$. Each iteration of the while-loop at line 7 (corresponding to a lift operation of f via v, followed by an update of the list L) costs $\mathcal{O}(|\mathsf{post}(v)| + |\mathsf{pre}(v)|)$. Since the value $f(v)$ for each vertex v can increase at most $\mathcal{M}_{G^\Gamma} + 1$ times, the global cost of Algorithm 1 is:

$$\mathcal{O}(\sum_{v \in V}(|\mathsf{post}(v)| + |\mathsf{pre}(v)|) \cdot \mathcal{M}_{G^\Gamma}) = \mathcal{O}(|E| \cdot \mathcal{M}_{G^\Gamma})$$

□

We are now ready to state the following theorem, relative to the complexity of the energy games problems introduced in Section 2.

Theorem 7. *Let $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$ be an* EG. *The decision problem, the strategy synthesis problem, the partition problem, and the minimum credit problem on Γ can be solved in time $\mathcal{O}(|E| \cdot \mathcal{M}_{G^\Gamma})$.*

Proof. It follows immediately from Lemma 6, Theorem 4, Theorem 5, and Theorem 6. □

Remark 1. Note that it is also possible to use the results in Section 3 and Section 4 (and in particular Lemma 5 and Lemma 6) to derive an algorithm that solves the decision problem for EG by reducing it to the decision problem for *safety games* [23]. A safety game is simple 2-player game played on an un-weighted arena, where the vertices are partitioned into allowed and forbidden positions. The goal of player 0 is that of building a play that never enters any forbidden position. It is well known that safety games can be solved in time linear w.r.t. the size of the corresponding arenas. On the ground of Lemma 5 and Lemma 6, the decision problem for an EG $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$ can be reduced to the decision problem for a safety game Γ' in time and *space* $\mathcal{O}(|E| \cdot \mathcal{M}_{G^\Gamma})$, i.e. pseudopolynomial w.r.t. the size of Γ. In fact, given the EG $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$, we can build a safety game Γ' played on an arena having set of position $V \times \{0, \ldots, \mathcal{M}_{G^\Gamma}\} \cup \{\top\}$. Given a vertex (v, k) in Γ', v represents a position in the original EG and k corresponds to a level of energy. The set of forbidden positions in the safety game Γ' is $\mathcal{B} = \{(v, \top) \mid v \in V\}$. The encoding safety game Γ' contains the edge $((v, k), (v', k'))$ iff:

$$(v, v') \in E \wedge k \neq \top \wedge (k + w(v, v') = k' \leq \mathcal{M}_{G^\Gamma} \vee (k + w(v, v') > \mathcal{M}_{G^\Gamma} \wedge k' = \top)).$$

It is easy to see that player 0 has a winning strategy from (v, k) in the encoding safety game Γ' (i.e. she can avoid to reach a (forbidden) position with global energy level $\top$) iff player 0 has a winning strategy from the vertex v with credit k in the original EG Γ.

Algorithm 1 can be also seen as a space-efficient counterpart of the above procedure of reduction from energy games to safety games where, rather than maintaining explicitly the whole space of possible energy values, we efficiently update online a set of energy-counters. This allows to use a space linear w.r.t. the size of the arena of the EG, rather than pseudopolynomial.

5 Solving the Mean-Payoff Game Problems

In this section we provide new efficient pseudopolynomial algorithms for the MPG problems stated in Section 2, featuring a better worst-case complexity than the corresponding state-of-the-art pseudopolynomial procedures by Zwick and Paterson [19]. Our new solutions for those problems build up on the notion of small energy progress measure and use Algorithm 1 as a basic step.

5.1 Decision Problem, Strategy Synthesis, and Three-Way Partition

First, we consider the decision problem and the strategy synthesis problem for MPG. Let $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ be a MPG where $w : V \to \{-W, \ldots, W\}$, and let $\nu \in \mathbb{Z}$. Consider the problem to decide if the value of a given vertex $v \in V$ is greater than or equal to ν. If $|\nu| > W$, then according to Lemma 2 we can immediately provide an answer to this decision problem (YES if $\nu < -W$, NO if $\nu > W$). Otherwise, consider the game $\Gamma^{-\nu} = (V, E, w - \nu, \langle V_0, V_1 \rangle)$, this game can be used to solve our original problems as stated in the following lemma.

Lemma 9. *Given a* MPG $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ *and a threshold* $\nu \in \mathbb{Z}$, *let* f *be a small energy progress measure for* $\Gamma^{-\nu} = (V, E, w - \nu, \langle V_0, V_1 \rangle)$. *All strategies* π_0^f *of player* 0 *compatible with* f *secure a payoff at least* ν *from all* $v \in V_f$ *in the* MPG Γ.

Proof. Towards contradiction, assume that $v \in V_f$ is a vertex such that the payoff that player 0 can secure from v is less than ν. By Lemma 1, the graph $G^\Gamma(\pi_0^f)$ admits a path $p = v_0 v_1 \ldots v_i \ldots v_n$ from $v = v_0$ to a cycle $v_i \ldots v_n$ ($v_i = v_n$) having average weight $\frac{1}{n-i} \sum_{j=i}^{n-1} w(v_j, v_{j+1}) < \nu$, which implies:

$$\sum_{j=i}^{n-1} (w - \nu)(v_j, v_{j+1}) < (n - i)\nu - (n - i)\nu = 0.$$

Since $v \in V_f$, the inductive application of the definition of f for $0 \leq j \leq n$ yields $v_j \in V_f$ for all $0 \leq j \leq n$. Hence, the graph $G^{\Gamma^{-\nu}}(\pi_0^f) \restriction V_f$ admits a path $v_0 \ldots v_i \ldots v_n$ from v to a negative cycle. This contradicts Lemma 4, since $f \restriction V_f$ is an energy progress measure on $G^{\Gamma^{-\nu}}(\pi_0^f) \restriction V_f$. $\square$

We now turn to the three-way partition problem, and we show how this problem can also be solved in time $\mathcal{O}(|E| \cdot \mathcal{M}_{G^\Gamma})$ using Algorithm 1 as a basic ingredient. In fact, consider the MPG $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$ where $w : V \to [-W, \dots, 0, \dots, +W]$, and define $V_* := V_0, V^* := V_1$. Given $\nu \in \mathbb{Z}, |\nu| \leq W$, we can construct the two game graphs $\Gamma' = (V, E, w - \nu, \langle V_0 := V_*, V_1 := V^*\rangle)$ and $\Gamma'' = (V, E, -w + \nu, \langle V_0 := V^*, V_1 := V_*\rangle)$. Running Algorithm 1 on Γ' yields the partition on V into $V_{\geq\nu}$ (for vertices securing player 0 a payoff at least ν in Γ) and $V_{<\nu}$. Running Algorithm 1 on Γ'' yields the partition on V into $V_{\leq\nu}$ and $V_{>\nu}$. The desired three-way partition can be immediately extracted from the above two partitions. Thus, we obtain:

Theorem 8. *Let $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$ be a* MPG. *The decision problem, the strategy synthesis problem, and the three-way partition problem on Γ can be solved in time $\mathcal{O}(|E| \cdot |V| \cdot W)$.*

Proof. For the decision problem and the strategy synthesis problem, the result immediately follows from Lemmas 2, 9, and Theorems 4, 6.

For the three-way partition problem, let Γ be a MPG with weight function w, and $\nu \in \mathbb{Z}$. Consider the game Γ' obtained as a copy of Γ with weight function $w - \nu$, and the game Γ'' obtained by exchanging the role of the players in Γ and with value function $-w + \nu$. By Theorem 1, Lemma 9, and Theorem 4, running Algorithm 1 on Γ' yields the partition $V_{\geq\nu}$ (for vertices from which player 0 secures a payoff at least ν in Γ) and $V_{<\nu}$, while on Γ'', it yields the partition on V into $V_{\leq\nu}$ and $V_{>\nu}$. Hence the three-way partition of V into $\langle V_{<\nu}, V_{=\nu} = V_{\leq\nu} \cap V_{\geq\nu}, V_{>\nu}\rangle$ can be hence obtained in time $\mathcal{O}(|E| \cdot \mathcal{M}_{G^\Gamma})$, by Theorem 6. $\square$

5.2 Value Problem and Optimal Strategy Synthesis

We finally consider the value problem. Let $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$ be a MPG. By Lemma 2, for each vertex $v \in V$, the value $\mathsf{val}^\Gamma(v)$ is contained in the following set of rationals:

$$S = \{\frac{p}{m} \mid p, m \in \mathbb{Z}, 1 \leq m \leq |V| \wedge -m \cdot W \leq p \leq m \cdot W\}.$$

Thus, a conceptually simple algorithm for computing the value of each vertex $v \in V$ would be to perform $|V|$ dichotomic searches in the set S. In particular, given $v \in V$ and $\frac{p}{m} \in S$, the application of Algorithm 1 on $\Gamma' = (V, E, m \cdot w - p, \langle V_0, V_1\rangle)$ allows to decide whether $\mathsf{val}^\Gamma(v) \geq \frac{p}{m}$. The global cost of such an algorithm is $\mathcal{O}(|E| \cdot |V|^3 \cdot W \cdot \log(|V| \cdot W)$, since S has size $\mathcal{O}(|V|^2 \cdot W)$ and Algorithm 1 is called on a reweighted version of Γ, where the maximal (absolute) weight is $2 \cdot |V| \cdot W$. In the sequel, we build on the ideas described above to design an $\mathcal{O}(|V|^2 \cdot |E| \cdot W \cdot \log(|V| \cdot W))$ algorithm to compute the values of a MPG.

Instead of performing a dichotomic search in S to assign the value $\mathsf{val}^\Gamma(v)$ to each vertex $v \in V$ individually, Algorithm 2 combines the dichotomic search with recursive calls. Each branch of the (binary) recursive tree for Algorithm 2 builds a sequence $\Gamma = \Gamma^0, \dots, \Gamma^n$ of game subgraphs of Γ, coupled with a decreasing sequence $S = S^0 \supseteq S^1 \supseteq \cdots \supseteq S^n$ of subsets of S such that:

- for all $0 \leq i \leq n$, the values of the vertexes in Γ^i are included in S^i
- for all $0 \leq i < n$, it holds $\max(S^{i+1}) - \min(S^{i+1}) \leq \frac{1}{2}(\max(S^i) - \min(Si))$

In particular, the second item above ensures that the length of each branch in the tree of recursive calls in Algorithm 2 is at most $\mathcal{O}(\log(V \cdot W))$, since the difference between two values in S is at most $2W$ and at least $\frac{1}{V^2}$. Each recursive call to Algorithm 2 on Γ^i, S^i, where S^i is represented by its extreme values $r_i = \min(S^i), s_i = \max(S^i)$, performs the following operations. First, it determines the largest element a_1 of S^i less than or equal to $\frac{r_i+s_i}{2}$, and the smallest element a_2 of S^i greater than or equal to[5] $\frac{r_i+s_i}{2}$. (e.g. by simply enumerating on the fly the elements in S^i). Then, Algorithm 1

[5] Note that $\frac{r_i+s_i}{2}$ is not guaranteed to be an element of S^i, since its denominator may not belong to the range $1 \dots |V|$.

is used to determine the partition $\langle V^i_{<a_1}, V^i_{=a_1}, V^i_{=a_2}, V^i_{>a_2}\rangle$ over the set of vertices V_i of the game subgraph Γ^i. Finally, Algorithm 2 is recursively called on the disjoint subgames $\Gamma^i \restriction V^i_{<a_1}, \Gamma^i \restriction V^i_{>a_2}$. The recursive reduction of the problem to smaller and disjoint instances provides a linear (w.r.t. $|V|$) improvement of Algorithm 2 over the naive iterative procedure.

The correctness of Algorithm 2 is established in Theorem 9 using the following lemma.

Lemma 10. *Given a* MPG $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$ *and* $\mu \in \mathbb{Q}$, *consider* $\Gamma' = \Gamma \restriction V_{\sim\mu}$, *where* $\sim \in \{<, >\}$. *If* $v \in V_{\sim\mu}$, *then* $\mathsf{val}^{\Gamma'}(v) = \mathsf{val}^{\Gamma}(v)$.

Proof. We start by showing that the relation $E \restriction V_{\sim\mu}$, where $\mu \in \mathbb{Q}$ and $\sim \in \{<, >\}$, is total[6]. Let π_0 (resp. π_1) be an optimal memoryless strategy in Γ for player 0 (resp. player 1), and consider the graph $G^\Gamma(\pi_0, \pi_1)$. Given $v \in V_{\sim\mu}$, consider the maximal (i.e cyclic) path from v in $G^\Gamma(\pi_0, \pi_1)$, $v_0, \dots, v_n, v_0 = v, v_n = v_k, 0 \le k \le n$. In such a path, the average weight of the cycle $v_k, \dots, v_n$ determines the payoff of each element $v_i, 0 \le i \le n$. Since π_0 and π_1 are optimal, for all $0 \le i \le n$, $\mathsf{val}^\Gamma(v_i)$ equals the payoff from v_i using π_0 (resp. π_1) against π_1 (resp. π_0). Hence, the average weight of the cycle $v_k, \dots, v_n$ is equal to $\mathsf{val}^\Gamma(v) \sim \mu$, and for all $1 \le i \le n$, $\mathsf{val}^\Gamma(v_i) \sim \mu$. In particular, v_1 is a successor of v having value $\sim \mu$, which implies that $E \restriction V_{\sim\mu}$ is total.

Given the above premise, the result follows from the fact that Γ' is a mean-payoff game and that for all memoryless optimal strategies $\pi_i \in \Sigma_i^M$ of player i ($i = 0, 1$), for all vertices $v \in V_i$, we have $\mathsf{val}^\Gamma(v) = \mathsf{val}^\Gamma(\pi_i(v))$. Therefore, all edges (v, v') in Γ such that $\mathsf{val}^\Gamma(v) \neq \mathsf{val}^\Gamma(v')$ are useless for optimality, and in particular, playing in Γ or in Γ' does not change the optimal value. □

Theorem 9. *Let* $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$ *be a mean-payoff game such that* $w : V \to \{-W, \dots, W\}$. *Algorithm 2 applied to the input* $(\Gamma, -W, W)$ *computes for each* $v \in V$ *the value* $\mathsf{val}^\Gamma(v)$.

Proof. Given the mean-payoff game $\Gamma = (V, E, w, \langle V_0, V_1\rangle)$, let $\{\mathsf{val}^\Gamma(v) \mid v \in V\} \subseteq U \subseteq S = \{\frac{p}{m} \mid 1 \le m \le |V| \wedge -W \le \frac{p}{m} \le W\}$. We prove that Algorithm 2$(\Gamma, \min(U), \max(U))$ terminates and computes the set of values in Γ. The termination is ensured by the fact that Algorithm 2 performs a binary search over the (finite) set $U \subseteq S$, where $|S| = \mathcal{O}(|V|^2 \cdot W)$. To complete the proof of our claim of correctness, we use an inductive argument on $|U|$.

For the base case, $|U| = 1$. Let $\frac{p}{q}$ be the only element of U such that for all $v \in V$, it holds $\mathsf{val}^\Gamma(v) = \frac{p}{q}$. By Lemma 9 and by Theorem 4, the application of Algorithm 1 to the game $\Gamma' = (V, E, qw - p, \langle V_0, V_1\rangle)$ (resp. $\Gamma'' = (V, E, -qw + p, \langle V_1, V_0\rangle)$) at Lines (5)–(6) yields the partition $\langle V_{\ge \frac{p}{q}} = V, V_{< \frac{p}{q}} = \emptyset\rangle$ (resp. $\langle V_{\le \frac{p}{q}} = V, V_{> \frac{p}{q}} = \emptyset\rangle$) on V, where $V_{\sim \frac{p}{q}} = \{v \in V \mid \mathsf{val}^\Gamma(v) \sim \frac{p}{q}\}, \sim \in <, \le, \ge, >$. Hence, Line (9) correctly assigns to each node of V its value $\mathsf{val}^\Gamma(v) = \frac{p}{q}$.

For the inductive step, suppose that $|U| > 1$. Let $a_1 = \max\{\frac{q}{l} \mid 1 \le l \le |V| \wedge \min(U) \le \frac{q}{l} \le \frac{1}{2}(\min(U) + \max(U))\}, a_2 = \min\{\frac{q}{l} \mid 1 \le l \le |V| \wedge \max(U) \ge \frac{q}{l} \ge \frac{1}{2}(\min(U) + \max(U))\}$. By Lemma 9 and by Theorem 4, the application of Algorithm 1 Lines (5)–(10) assign to each node in $V_{=a_1}$ (resp. $V_{=a_2}$) its value $\mathsf{val}^\Gamma(v) = a_1$ (resp. $\mathsf{val}^\Gamma(v) = a_2$). By inductive hypothesis and by Lemma 10 Line (13) (resp. Line (14)) computes the value of the nodes in $V_{<a_1}$ (resp. $V_{>a_2}$). □

Lemma 11 proves that the height of the tree of recursive calls corresponding to Algorithm 2 is asymptotically logarithmic w.r.t. $|V|^2 \cdot W$, and Lemma 12 states that the cost of executing lines (1)–(11) in each recursive call of Algorithm 2 for the subgame $\Gamma' = (V', E', w, \langle V_0', V_1'\rangle)$ is $\mathcal{O}(|E'| \cdot |V'|^2 \cdot W)$. Here, the quadratic dependence on $|V'|$ comes from the need of applying Algorithm 1 (cf. Lines (5)–(8)) on a reweighted version of Γ', where all edge-weights are multiplied by a natural number of size at most $|V'|$. Theorem 10 then gives the complexity of Algorithm 2 for the value problem on mean-payoff games.

[6] i.e. $\forall v \in V_{\sim\mu} \exists u \in V_{\sim\mu}((v, u) \in E \restriction V_{\sim\mu})$

Algorithm 2: Solving the value problem for mean-payoff games.

Input : Mean-payoff game $\Gamma = (V, E, w : V \to \{-W, \dots, W\}, \langle V_0, V_1 \rangle)$; lower and upper bounds $\frac{p_1}{m_1} \leq \frac{p_2}{m_2}$ on the values of the nodes in Γ, where $-W \leq \frac{p_1}{m_1} \leq \frac{p_2}{m_2} \leq W$, $p_1, p_2, m_1, m_2 \in \mathbb{N}$, and $1 \leq m_1 \leq |V|, 1 \leq m_2 \leq |V|$.

Output : For each $u \in V$, the value $v^{\Gamma}(u)$.

begin

1 **if** $V \neq \emptyset$ **then**

2 $a_1 \leftarrow \frac{q_1}{l_1} \leftarrow \max\{\frac{q}{l} \mid 1 \leq l \leq |V| \wedge \frac{p_1}{m_1} \leq \frac{q}{l} \leq \frac{1}{2}(\frac{p_1}{m_1} + \frac{p_2}{m_2})\}$

3 $a_2 \leftarrow \frac{q_2}{l_2} \leftarrow \min\{\frac{q}{l} \mid 1 \leq l \leq |V| \wedge \frac{p_2}{m_2} \geq \frac{q}{l} \geq \frac{1}{2}(\frac{p_1}{m_1} + \frac{p_2}{m_2})\}$

4

/ Use Algorithm 1 to determine $V_{<a_1}, V_{=a_1}, V_{=a_2}, V_{>a_2}$ */*

5 $f_1 \leftarrow$ Algorithm 1$(V, E, l_1 w - q_1, \langle V_0, V_1 \rangle)$

6 $f_2 \leftarrow$ Algorithm 1$(V, E, -l_1 w + q_1, \langle V_1, V_0 \rangle)$

7 $f_3 \leftarrow$ Algorithm 1$(V, E, l_2 w - q_2, \langle V_0, V_1 \rangle)$

8 $f_4 \leftarrow$ Algorithm 1$(V, E, -l_2 w + q_2, \langle V_1, V_0 \rangle)$

9 **foreach** $(u \mid f_1(u) \neq \top \wedge f_2(u) \neq \top)$ **do** $v^{\Gamma}(u) \leftarrow a_1$

10 **foreach** $(u \mid f_3(u) \neq \top \wedge f_4(u) \neq \top)$ **do** $v^{\Gamma}(u) \leftarrow a_2$

11 $V_{<a_1} \leftarrow \{u \mid f_1(u) = \top\}$; $V_{>a_2} \leftarrow \{u \mid f_4(u) = \top\}$

12

*/*Recursive Calls*/*

13 Algorithm 2$((V_{<a_1}, E \upharpoonright V_{<a_1}, w \upharpoonright V_{<a_1}, \langle V_0 \cap V_{<a_1}, V_1 \cap V_{<a_1} \rangle), \frac{p_1}{m_1}, a_1)$

14 Algorithm 2$((V_{>a_2}, E \upharpoonright V_{>a_2}, w \upharpoonright V_{>a_2}, \langle V_0 \cap V_{>a_2}, V_1 \cap V_{>a_2} \rangle), a_2, \frac{p_2}{m_2})$

end

Lemma 11. *Let $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ be a* MPG, *where $w : V \to \{-W, \dots, W\}$. The height of the tree of recursive calls corresponding to Algorithm 2 applied to $(\Gamma, -W, W)$ is $\mathcal{O}(\log(|V| \cdot W))$.*

Proof. Let $c_1, \dots, c_k$ be a branch of recursive calls in the tree of recursive calls corresponding to Algorithm 2, and let $d_1, \dots, d_k$ be the distances between the input parameters $\frac{p_1}{m_1}, \frac{p_2}{m_2}$ in each recursive call $c_1, \dots, c_k$. Then, $d_1 = 2W$ and for all $i = 1, \dots, k$, $d_{i+1} \leq \frac{d_i}{2}$. Since two rational numbers with denominator at most $|V|$ have distance at least $\frac{1}{|V|^2}$, we obtain that $k \leq \log \frac{2W}{|V|^{-2}} = \log(2W|V|^2)$. □

Lemma 12. *The cost of executing lines* (1)–(11) *in a recursive call of Algorithm 2 on the subgame $\Gamma' = (V', E', w, \langle V'_0, V'_1 \rangle)$ of $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$, where $w : V \to \{-W, \dots, W\}$, is $\mathcal{O}(|E'| \cdot |V'|^2 \cdot W)$.*

Proof. By assumption, $1 \leq m_{i \in \{1,2\}} \leq |V'|$, and $-|V'| \cdot W \leq p_{i \in \{1,2\}} \leq |V'| \cdot W$. Hence, the codomain of the functions $m_1 w - p_1, -m_1 w + p_1, m_2 w - p_2, -m_2 w + p_2$ is the set of integers $\{-2 \cdot |V'| \cdot W, \dots, 2 \cdot |V'| \cdot W\}$. By Theorem 6, Lines (5)–(8) can be executed in time $\mathcal{O}(|E'| \cdot |V'|^2 \cdot W)$. Lines (9)–(11) cost $\mathcal{O}(|V'|)$, while Lines (2)–(3) can be trivially executed in time $\mathcal{O}(|V'|^2 \cdot W)$ (e.g., by simply enumerating the elements of the set $S = \{\frac{p}{m} \mid 1 \leq m \leq |V'| \wedge -W \leq \frac{p}{m} \leq W\}$, where $|S| = \mathcal{O}(|V'|^2 \cdot W)$). □

Theorem 10. *Let $\Gamma = (V, E, w, \langle V_0, V_1 \rangle)$ be a* MPG *where $w : V \to \{-W, \dots, W\}$. Algorithm 2 applied to $(\Gamma, -W, W)$ solves the value problem and the optimal strategy synthesis problem on Γ in time: $\mathcal{O}((\log(|V|) + \log(W)) \cdot |E| \cdot |V|^2 \cdot W)$.*

Proof. Let $\Gamma^{\langle \ell, 1 \rangle}, \dots, \Gamma^{\langle \ell, k_\ell \rangle}$ be the k_ℓ subgames of Γ considered at level ℓ of the tree of recursive calls corresponding to Algorithm 2. Then, the corresponding set of vertices $V^{\langle \ell, 1 \rangle}, \dots, V^{\langle \ell, k_\ell \rangle}$ are such that $V^{\langle \ell, i \rangle} \cap V^{\langle \ell, j \rangle} = \emptyset$ for $j \neq i, 1 \leq i, j, \leq k_\ell$, and $V^{\langle \ell, 1 \rangle} \cup \dots \cup V^{\langle \ell, k_\ell \rangle} \subseteq V$. Hence, on the ground of Lemmas 11, 12 we obtain that the global complexity of Algorithm 2 is:

$$\mathcal{O}(\sum_{\ell=1}^{\log(2|V|^2W)} |E|\cdot|V^{\langle\ell,1\rangle}|^2\cdot W+\cdots+|E|\cdot|V^{\langle\ell,k_\ell\rangle}|^2\cdot W)=$$

$$=\mathcal{O}((\log(|V|)+\log(W))\cdot|E|\cdot|V|^2\cdot W)$$

since $|E|\cdot|V^{\langle\ell,1\rangle}|^2\cdot W+\cdots+|E|\cdot|V^{\langle\ell,k_\ell\rangle}|^2\cdot W \leq |E|\cdot W\cdot(|V^{\langle\ell,1\rangle}|+\cdots+|V^{\langle\ell,k_\ell\rangle}|)^2 \leq |E|\cdot|W|\cdot|V|^2$. □

Thus, our MPG value problem outperforms the corresponding deterministic procedure in [19], when the maximum weight in the MPG graph is small (i.e. W is subexponential w.r.t. $|V|$). To design a MPG value algorithm that outperforms previous solutions for all values of W, we can consider a *randomized* framework and combine our procedure with the one proposed in [3]. In particular, the solution to the value problem proposed by [3] has expected complexity $|V|^2\cdot|E|\cdot e^{2\cdot\sqrt{|V|\cdot\ln(|E|/\sqrt{|V|})}+O(\sqrt{|V|}+\ln|E|)}$. By interleaving our MPG value algorithm with [3] and adding a stopping criterion which terminates the computation when either of the two procedures finishes, we get a randomized algorithm for the MPG value problem with expected complexity $\min(\mathcal{O}(|V|^2\cdot|E|\cdot W\cdot\log(|V|\cdot W)), |V|^2\cdot|E|\cdot e^{2\cdot\sqrt{|V|\cdot\ln(|E|/\sqrt{|V|})}+O(\sqrt{|V|}+\ln|E|)})$, which outperforms all previous solutions.

6 Conclusion

We designed simple and efficient deterministic algorithms for solving energy games and mean-payoff games. Our algorithmic engine requires $\mathcal{O}(|E|\cdot|V|\cdot W)$ computational steps to solve the MPG decision problem, outperforming the corresponding $\Theta(|E|\cdot|V|^2\cdot W)$ pseudopolynomial procedure in [19]. Note that the algorithm in [19] requires *always* $\Theta(|E|\cdot|V|^2\cdot W)$, while our procedure is $\mathcal{O}(|E|\cdot|V|\cdot W)$ only in the worst case (it needs linear time when, for example, all the weights are positive). The value problem can be solved in time $\mathcal{O}(|E|\cdot|V|^2\cdot W(\log|V|+\log W))$ using our framework, while [19] requires $\Theta(|E|\cdot|V|^3\cdot W)$. As [12], our algorithm has also the advantage to produce as a byproduct (optimal) winning strategies, while [19] needs further computation for strategy synthesis. Hence, the winning strategy synthesis problem (resp. the optimal strategy synthesis problem) is solved in time $\mathcal{O}(|E|\cdot|V|\cdot W)$ (resp. $\mathcal{O}(|E|\cdot|V|^2\cdot W(\log|V|+\log W))$) using our procedures, outperforming [12, 19]. In combination with the randomized algorithm of Andersson and Vorobyov [3], our MPG value algorithm is a randomized procedure with currently the best expected complexity, namely:

$$\min(O(|V|^2\cdot|E|\cdot W\cdot\log(|V|\cdot W)), |V|^2\cdot|E|\cdot e^{2\cdot\sqrt{|V|\cdot\ln(|E|/\sqrt{|V|})}+O(\sqrt{|V|}+\ln|E|)})$$

References

1. A. Ehrenfeucht and J. Mycielski. International journal of game theory. *Positional Strategies for Mean-Payoff Games*, 8:109–113, 1979.
2. A. V. Karzanov and V. N. Lebedev. Cyclical games with proibitions. *Mathematical Programing*, 60:277–293, 1993.
3. D. Andersson and S. Vorobyov. Fast algorithms for monotonic discounted linear programs with two variables per inequality. Technical Report Preprint NI06019-LAA, Isaac Newton Institute for Mathematical Sciences, Cambridge, UK, 2006.
4. P. Bouyer, U. Fahrenberg, K. G. Larsen, N. Markey, and J. Srba. Infinite runs in weighted timed automata with energy constraints. In *Proc. of FORMATS: Formal Modeling and Analysis of Timed Systems*, LNCS 5215, pages 33–47. Springer, 2008.

5. A. Chakrabarti, L. de Alfaro, T. A. Henzinger, and M. Stoelinga. Resource interfaces. In *Proc. of EMSOFT: Embedded Software*, LNCS 2855, pages 117–133. Springer, 2003.
6. J. Chaloupka and L.Brim. Faster algorithm for mean-payoff games. In *Proc. of the 5-th Docoral Workshop on Mathematical and Engineering Methods in Computer Science (MEMICS '09)*, pages 45–53. NOVPRESS, 2009.
7. A. Condon. On algorithms for simple stochastic games. In *Advances in Computational Complexity Theory*, volume 13 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 51–73. American Mathematical Society, 1993.
8. V. Dhingra and S. Gaubert. How to solve large scale deterministic games with mean payoff by policy iteration. In *Proc. Performance evaluation methodolgies and tools,* article no. 12. ACM, 2006.
9. L. Doyen, R. Gentilini, and J.-F. Raskin. Faster pseudopolynomial algorithms for meanpayoff games. Technical Report 2009.120, Universite Libre de Bruxelles (ULB), Bruxelles, Belgium, 2009.
10. E. A. Emerson, C. Jutla, and A. P. Sistal. On model checking for fragments of the μ-calculus. In *Proc. of CAV: Computer Aided Verification*, LNCS 697, pages 385–396. Springer, 1993.
11. Y. Gurevich and L. Harrington. Trees, automata, and games. In *Proc. of STOC: Symposium on Theory of Computing*, pages 60–65. ACM, 1982.
12. H. Bjorklund and S. Vorobyov. A combinatorial strongly subexponential strategy improvement algorithm for mean payoff games. *Discrete Applied Mathematics*, 155:210–229, 2007.
13. M. Jurdzinski. Deciding the winner in parity games is in UP $\cap$ coUP. *Inf. Process. Lett.*, 68(3):119–124, 1998.
14. M. Jurdzinski. Small progress measures for solving parity games. In *Proceedings of STACS: Theoretical Aspects of Computer Science*, LNCS 1770, pages 290–301. Springer, 2000.
15. D. Kozen. Results on the propositional mu-calculus. *Theor. Comput. Sci.*, 27:333–354, 1983.
16. N. Pisaruk. Mathematics of operations research. *Mean Cost Cyclical Games*, 4(24):817–828, 1999.
17. C. M. Papadimitriou. *Computational complexity*. Addison-Wesley, Reading, Massachusetts, 1994.
18. S. Schewe. From parity and payoff games to linear programming. In *Proceedings of MFCS: Mathematical Foundations of Computer Science*, LNCS 5734, pages 675–686. Springer, 2009.
19. U. Zwick and M. Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158:343–359, 1996.
20. V. A. Gurvich, A. V. Karzanov, and L. G. Kachiyan. Ussr computational mathematics and mathematical physics. *Cyclic Games and an Algorithm to Find Minmax Cycle Means in Directed Graphs*, 5(28):85–91, 1988.
21. I. Walukiewicz. Pushdown processes: Games and model checking. In *Proceedings of CAV: Computer Aided Verification*, LNCS 1102, pages 62–74. Springer, 1996.
22. Y. Lifshits and D. Pavlov. Potential theory for mean payoff games. *Journal of Mathematical Sciences*, 145(3):4967–4974, 2007.
23. Z. Manna and A.Pnueli. *The Temporal Logic of Reactive and Concurrent Programs*. Springer-Verlag, 1992.